Code Simplicity: Software Design In Open Source Projects

Max Kanat-Alexander
max@codesimplicity.com

Before The Show

- I talk quickly
- Stop me if you don't understand
  - Also stop me if you need examples
  - Questions or disagreements at end
- Points will be fast, but still important
  - More detail on everything at my blog: http://www.codesimplicity.com/
- Some of these things you may already know, what matters is pointing them out as important.

- Who am I?
  - Assistant Project Lead for Bugzilla
  - Writer of Code Simplicity
  - Long-time programmer and system administrator
- Data comes from:
  - Experience programming and designing
    - Bugzilla “experiment”
  - Interviewing many programmers
  - Reading extensively with analysis
    - Data collection is difficult because of the timeframe of software projects.
- I speak very definitely, but please make up your own mind about the things I am saying.

What Is Software Design?

- Administrative Decisions
  - What programmer to put where
  - Development timeframes
  - etc.
- Coding
- **Technical Decisions**
  - What language to use
  - What technologies to choose
  - etc.

There Is No Science of Software Design

- Science requires:
  - Laws
  - Proof
  - Results
- Many methodologies, no science:
  - “Waterfall”
  - “Agile”
  - etc.

- Not going to prove anything today, just show
  - Can prove, though, in various ways
- There are similar ideas out there, but they are not the same as what I am going to talk about
  - Largely they are not low-level enough
    - Seven Principles: http://c2.com/cgi/wiki?SevenPrinciplesOfSoftwareDevelopment
  - They tell you *what* to do, I only help you make decisions for yourself and try to tell you *why*.
  - I did not derive from any of these methodologies, but the bits of them that work could be derived from what I am going to tell you.

Why Have a Science of Software Design?

- Help Make Technical Decisions
- *Why* do some things “work” and others don't?

Results

- Bugzilla
- Improved My Own Programming
  - Resolved every question
- Brought novices to understand *why*
- Explained difficulties and “war stories” of experienced programmers

Not Brainwashing or Marketing

- Not going to tell you what decisions to make, just going to give you information that will help you make them
  - This differs from methodologies
- Buzzword-free

FOSS vs. Proprietary

- The basics are the same, but application can be different.

Purpose Of Software

- **To help people**
  - Never “to help the computer”
  - Specific software is “To help people (blah)”
- Stated purpose should be:
  - Short
  - Simple
  - Specific
  - Needed
  - Followed Exactly

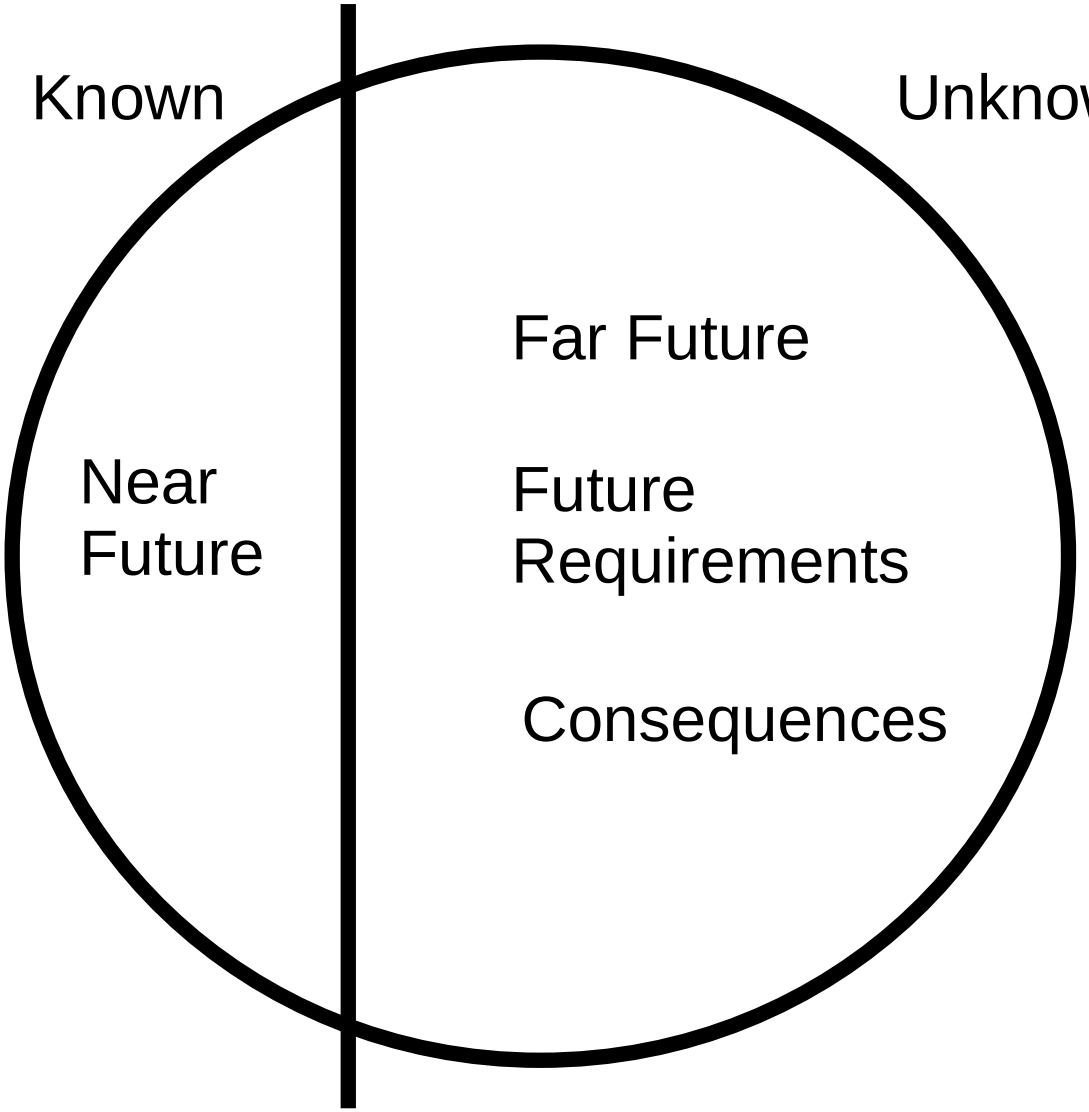
Goals of Software Design

- Be as helpful as possible
- Continue to be as helpful as possible
- Make decisions that make it easy to be (and continue being) as helpful as possible

Primary Law: Future

- There is more future than present
  - Future is composed of infinite series of presents
- The future is more important than the present
- Effort spent on design should be proportional to how much future time there is in which you expect the software to exist
  - Planning to re-write is unnecessary

Future: Known vs. Unknown



Software has long time lines

Law Of Change

- The longer your software exists, the more probable it is that any piece of it will have to change
  - Means that as time goes on, *every* piece is likely to change

Law of Defect Probability

- The chance of introducing a defect into your program is directly proportional to the size of the changes you make
  - Perfection is impossible
- Write as little code as possible
- Don't fix what isn't broken
- Explains re-use

Law Of Simplicity

- The ease of maintenance of any piece of software is directly proportional to the simplicity of the individual pieces
  - Not of the whole system, just the individual pieces
- Stated differently, is inverse to the complexity.
- Simplicity is relative, largely to viewpoint
- How simple do you have to be?
  - Perspective of another programmer who's never seen your code
- Be consistent

What Is a Bug?

- Programmer's Intentions
  - Uncertainties can be resolved by:
    - Comments
    - Spec
    - “Reasonable programmer”
    - Assume he/she meant to do what is best for the user
- User Expectations
  - Specs that violate user expectations are spec bugs
  - If there's a conflict, it's “majority rules”
    - You can also add a preference, but that adds complexity

Where Do Bugs Come From?

- Complexity
  - The box with a million unlabeled buttons
- Misunderstandings
  - Particularly of language words, symbols, functions, etc.

“Law” of Testing

- You don't know it works unless you've tried it.

Application in FOSS

- **Must be more hardcore**
- Largely problems with geographic distribution
- Here's where I tell you some things to do, but you should still make up your own mind.

Difficulties of Design in FOSS

- Speed of change can be limited
  - Reviews
  - Checkin Procedures
  - Lack of Somebody to Talk To (IRC helps)
- Time available can be limited
  - Designer's Time
    - Have to communicate designs quickly
  - Implementer's Time
    - Have to be able to read design quickly

- Communication bandwidth is limited
  - Have to type to communicate design
  - No whiteboard, etc.
  - Group probably won't all be there at once
- Novices
  - Either to development in general or just your project.
- Desire for consistency vs. desire for development speed
- Disconnection with users
- User requirements = My requirements?

- Developers pick what to work on
  - May not want to conform to design
  - May not want to work on refactoring
- Getting out releases for testing

Solutions

- Extreme consistency
  - If you can't communicate a design, it helps if the existing code already works the right way
- Brief communications
- Code reviews
- Extensive developer documentation
- Lots of attention to newbies
  - Be nice

- Read support mail
- Read blogs about your product
  - But don't ever let your detractors write your requirements.
- Survey your users
- Have somebody who loves refactoring

The End: Q & A

http://www.codesimplicity.com/

max@codesimplicity.com